Signed MSB-Set Comb Method for Elliptic Curve Point Multiplication

Min Feng[1], Bin B. Zhu[2], Cunlai Zhao[1], and Shipeng Li[2]

[1] School of Mathematical Sciences, Peking Univ., Beijing, 100871, China
{fengmin, zhao}@math.pku.edu.cn

[2] Microsoft Research Asia, Beijing, 100080, China
{binzhu, spli}@microsoft.com

Abstract. Comb method is an efficient method to calculate point multiplication in elliptic curve cryptography, but vulnerable to power-analysis attacks. Various algorithms have been proposed recently to make the comb method secure to power-analysis attacks. In this paper, we present an efficient comb method and its Simple Power Analysis (SPA)-resistant counterpart. We first present a novel comb recoding algorithm which converts an integer to a sequence of signed, MSB-set comb bit-columns. Using this recoding algorithm, the signed MSB-set comb method and a modified, SPA-resistant version are then presented. Measures and precautions to make the proposed SPA-resistant comb method resist all power-analysis attacks are also discussed, along with performance comparison with other comb methods. We conclude that our comb methods are among the most efficient comb methods in terms of number of precomputed points and computational complexity.

1 Introduction

Elliptic curve cryptography (ECC) has gained increasing popularity in public key cryptography due to its shorter key sizes for the same level of security as compared to other public key cryptosystems. A key operation in ECC is point multiplication. Many efficient point multiplication methods have been developed [1]. One of them is the comb method [2]. The main idea is to use a binary matrix with rows and columns to represent a scalar and process the matrix columnwise. Unfortunately, the method is vulnerable to side-channel attacks which were first introduced by Kocher et al. [3,4] and extended to ECC [5]. Side-channel attacks measure observable parameters such as timings or power consumptions during cryptographic operations to deduce the whole or partial secret information of a cryptosystem. Power analysis includes both Simple Power Analysis (SPA) and Differential Power Analysis (DPA) [4]. A particular target of side-channel attacks for ECC is the scalar in point multiplication which computes a product kP where P is point on an elliptic curve $E(F)$ over a finite field F and k is a secret multiplier which is a positive integer. Higher order and refined DPA attacks are also proposed [6,7,8]. With power analysis, partial information or the exact value of the secret k can be deduced when the original comb method [2] or the scalar multiplication methods described in [1] are used.

Many countermeasures have been proposed to protect against side-channel attacks on ECC. Two major strategies have been proposed to protect against SPA attacks. The first strategy is to make the addition and doubling operations indistinguishable. A unified formula for computing both addition and doubling has been proposed in [9] for Jacobi-type and in [10] for Hesse-type elliptic curves. The second strategy is to remove dependency in the intermediate steps of the scalar multiplication on specific value of the secret multiplier k. Coron, et al. [5, 11, 12, 13, 14] proposed schemes using addition chains to always execute point addition and doubling for each bit. Möller, et al. [15, 16] modified window methods by making addition chains with fixed pattern of nonzero digits. Hedabou et al. proposed SPA-resistant comb methods [17, 18, 19]. Chevallier-Mames et al. [20] proposed a scheme which divides point doubling and point addition into side-channel atomic blocks so that point multiplication appears as a succession of side-channel atomic blocks that are indistinguishable by SPA. An SPA-resistant method is not necessarily resistant to DPA attacks. Many countermeasures have been proposed to convert an SPA-resistant method into a DPA-resistant method. Coron [5] proposed to use random projective coordinates. Joye and Tymen [21] proposed to use a random isomorphism such as a random elliptic curve isomorphism and a random field isomorphism.

In this paper, we propose a new comb recoding algorithm to convert each bit-column in the comb scalar matrix to a signed, Most Significant Bit (MSB)-set, nonzero bit-column. All nonzero bits in an arbitrary bit-column have the same sign. Using the recoding algorithm, we present a novel comb method which computes point multiplication more efficiently with less precomputed points than the original comb method [2]. The proposed comb method is then modified to be SPA-resistant by exploiting the fact that point addition and point subtraction are virtually of the same computational complexity in ECC and cannot be distinguished by SPA. We also describe measures to convert our SPA-resistant comb method to thwart all known side-channel attacks. Our comb methods are among the most efficient comb methods in terms of number of precomputed points and computational complexity.

This paper is organized as follows. In the next section, we introduce preliminaries for ECC and the original comb method. In Section 3, side-channel attacks and prior countermeasures are presented. Our novel comb recoding algorithm and comb point multiplication methods are described in Section 4. Security analysis and performance comparison with other comb methods are also provided in this section. We conclude this paper in Section 5.

2 Preliminaries

2.1 Elliptic Curves Equations

An elliptic curve over a field F can be expressed by its Weierstrass form:

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \quad a_i \in F.$$

The set $E(F)$ of points $(x, y) \in F^2$ satisfying the above equation plus the "point at infinity" $\mathcal{O}$ forms an abelian group with the point at infinity $\mathcal{O}$ as the zero, and point addition as the group's binary operation. Given two points P_1 and P_2 in $E(F)$, a third point $P_3 = P_1 + P_2 \in E(F)$ as the addition of P_1 and P_2 can be calculated with the chord-tangent process [1]. A special point addition that a point adds itself is called *doubling*. The cost of point doubling is usually different from that of point addition. Point addition and doubling need to compute costly field inversions. By using the Jacobian projective coordinates which represent a point $P = (x, y)$ as $P = (X, Y, Z)$, where $x = X/Z^2$ and $y = Y/Z^3$, and the infinity point $\mathcal{O}$ as $(\theta^2, \theta^3, 0)$, $\theta \in F^*$, field inversions can be avoided at the expense of more field multiplications. A field multiplication is usually much faster than a field inversion, resulting in faster elliptic curve point addition and doubling.

The group $E(F)$ generated by an elliptic curve over some finite field F meets the public key cryptography requirements that the discrete logarithm problem is very difficult to solve. Therefore ECC has been used in many standards and applications. Elliptic curves used in cryptography are elliptic curves defined over fields F_{2^m} or fields F_p where m is a large number and p is a big prime. Over these two types of fields, the Weierstrass form reduces to the short Weierstrass form, and point addition and doubling are also simplified. For details of elliptic curve equations and point operations, interested readers are referred to [1].

Algorithm 1. *Fixed-base Comb Method [2]* ($d = \lceil \frac{n}{w} \rceil$)

```
Input: A point $P$, an integer $k = \sum_{i=0}^{n-1} b_i 2^i$ with $b_i \in \{0,1\}$, and a window
       width $w \geqslant 2$.
Output: $Q = kP$.
Precomputation Stage:
1. Compute $[b_{w-1}, \cdots, b_1, b_0]P$ for all $(b_{w-1}, \cdots, b_1, b_0) \in \{0,1\}^w$.
2. Write $k = K^{w-1} || \cdots || K^1 || K^0$, where each $K^j$ is a bit-string of length $d$.
   Padding with 0 on the left if necessary. Let $K_i^j$ denote the $i^{th}$ bit of $K^j$. Define
   $\mathbb{K}_i \equiv [K_i^{w-1}, \cdots, K_i^1, K_i^0]$.
3. $Q = \mathcal{O}$.
Evaluation Stage:
4. For $i = d-1$ to 0 by $-1$ do:
5.     $Q = 2Q$,
6.     $Q = Q + \mathbb{K}_i P$.
7. Return $Q$.
```

2.2 Scalar Multiplication

Adding a point P to itself k times is called *scalar multiplication* or *point multiplication*, and is denoted as $Q = kP$, where k is a positive integer. Many efficient methods have been proposed for scalar multiplication. Interested readers are referred to [1] for details. One of the proposed efficient point multiplication methods is the comb method proposed by Lim and Lee [2] in 1994.

Let $k = \sum_{i=0}^{n-1} b_i 2^i$ with $b_i \in \{0,1\}$. For an integer $w \geqslant 2$, set $d = \lceil \frac{n}{w} \rceil$. We define $[b_{w-1}, b_{w-2}, \cdots, b_1, b_0] \triangleq b_{w-1}2^{(w-1)d} + b_{w-2}2^{(w-2)d} + \cdots + b_1 2^d + b_0$, where $(b_{w-1}, b_{w-2}, \cdots, b_1, b_0) \in \{0,1\}^w$. The comb method uses a binary matrix of w rows and d columns to represent an integer k, and processes the matrix columnwise.

This comb method stores $2^w - 1$ points in the precomputation stage. In storage estimation in this paper, the input point P is always included. Let us estimate the time cost of the comb method. In the precomputation stage, $[b_{w-1}, \cdots, b_1, b_0]P$ needs to be calculated for $(b_{w-1}, \cdots, b_1, b_0) \in \{0,1\}^w$. To achieve this, $2^d P, 2^{2d} P, \cdots, 2^{(w-1)d} P$ are first calculated, which costs $(w-1)d$ doubling operations. Then every possible combination of l $(l > 1)$ nonzero bits in $[b_{w-1}, \cdots, b_1, b_0]P$ is calculated by adding one point from P's point multiplication of $l-1$ bits combinations and a point from P's point multiplication of a single bit. Therefore it costs $2^w - w - 1$ point additions in the precomputation stage. In conclusion, the total cost in the precomputation stage is $\{(w-1)d\}D + \{(2^w - w - 1)\}A$.

To estimate the time cost in the evaluation stage, we assume the most significant column of $\{\mathbb{K}_i\}$ is not zero, i.e., $\mathbb{K}_{d-1} \neq 0$. Then the number of doubling operations in the evaluation stage is $(d-1)$. If $\mathbb{K}_i = 0$, then the point addition in Step 6 is not needed. If we assume k is uniformly distributed, the probability that $\mathbb{K}_i \neq 0$ is $\frac{2^w-1}{2^w}$, and the average number of point additions is $\frac{2^w-1}{2^w}(d-1)$. Therefore the average time cost in the evaluation stage is approximately $\{(d-1)\}D + \{\frac{2^w-1}{2^w}(d-1)\}A$. The total time cost of the comb method is $\{(w-1)d + (d-1)\}D + \{(2^w - w - 1) + \frac{2^w-1}{2^w}(d-1)\}A = \{wd-1\}D + \{(2^w - w - 1) + \frac{2^w-1}{2^w}(d-1)\}A$.

3 Side-Channel Attacks and Countermeasures

3.1 Side-Channel Attacks

Two types of power analysis have been introduced by P. Kocher [3, 4]. One is the Simple Power Analysis (SPA). The other is the Differential Power Analysis (DPA).

Simple Power Analysis. SPA analyzes a single trace of power consumption in a crypto-device during scalar multiplication. A branch instruction condition can be identified from the recorded power consumption data. This represents continuity of elliptic curve doubling operation. For the comb method Alg. 1, SPA can detect if $\mathbb{K}_i$ is zero or not, which means leak of secret information.

Differential Power Analysis. DPA records many power traces of scalar multiplications, and uses correlation among the records and error correction technique [4] to deduce some or all digits of the secret k. DPA is more complex yet powerful than SPA. An SPA-resistant scalar multiplication method is not necessarily resistant to DPA attacks, but many countermeasures can be used to transform an SPA-resistant method to a DPA-resistant method. A common practice is to make execution, and thus power consumption, different for identical inputs. Randomization is usually employed to achieve this effect. All these randomizing approaches are feasible: randomizing input point in projective coordinates, randomizing exponential parameter representation, randomizing elliptic curve equation, and randomizing field representation. This paper focuses on SPA-resistant scalar multiplication. All these randomizing approaches can be applied to transform our SPA-resistant methods to be resistant to DPA attacks.

3.2 Prior SPA-Resistant Comb Methods

Many countermeasures to SPA attacks have been proposed. A particular approach is to make execution of scalar multiplication independent of any specific value of the multiplier k. All the proposed SPA-resistant comb methods as well as the one to be proposed in this paper are of this approach. Those SPA-resistant comb methods are described next.

HPB's Comb Methods. Hedabou, Pinel and Bénéteau (HPB) [17, 18] proposed two comb methods recently to protect against SPA. The main idea is to extend $\mathbb{K}_i$ in the comb method Alg. 1 to a signed representation $(\mathbb{K}'_i, s_i)$, where each $\mathbb{K}'_i$ is nonzero.

Their first method [17] uses this following procedure to generate such a signed representation $(\mathbb{K}'_i, s_i)$ for an odd integer k represented by $\mathbb{K}_i, 0 \leqslant i < d$, in the comb method. Let $s_0 = 1$ and construct the rest by setting

$$\begin{cases} (\mathbb{K}'_i, s_i) & = (\mathbb{K}_{i-1}, s_{i-1}) \\ (\mathbb{K}'_{i-1}, s_{i-1}) & = (\mathbb{K}_{i-1}, -s_{i-1}) \end{cases}$$

if $\mathbb{K}_i = 0$, and $(\mathbb{K}'_i, s_i) = (\mathbb{K}_i, s_i)$ otherwise.

The second method [18] translates an odd scalar k into a representation $\sum_{i=0}^{n} b'_i 2^i$ with $b'_i \in \{1, -1\}$ by exploiting the facts $1 \equiv 1\bar{1}\bar{1}\cdots\bar{1}$, where $\bar{1}$ is defined as -1, and applies the original comb method to the new representation of the old scalar k. A bit-column $\mathbb{K}_i$ generated in this method can be represented by $[b_{w-1}, \cdots, b_1, b_0]$, where $b_j \in \{1, -1\}$, $0 \leqslant j < w$.

HPB's comb methods apply these signed representations to the original comb method to calculate $(k+1)P$ for even k and $(k+2)P$ for odd k. $2P$ is then calculated. P or $2P$ is subtracted from the result produced by the original comb method to obtain the desired point kP. HPB's first method will be referred to as the *signed Non-Zero* (sNZ) comb method, and the second method as the *signed All-Bit-Set* (sABS) comb method in this paper.

sNZ has the same time and space cost as the original comb method in the precomputation stage, i.e., storage of $2^w - 1$ points and time cost of $\{(w-1)d\}D + \{(2^w - w - 1)\}A$. Because elliptic curve point substraction has the same complexity as point addition, the second method stores only 2^{w-1} precomputed points $[b_{w-1}, b_{w-2}, \cdots, b_2, b_1, 1]P$ since the scalar is odd, where $b_i \in \{1, -1\}$.

The number of precomputed points in sABS is about half of that in sNZ. The time cost of sABS in the precomputation stage was estimated as $\{(w-1)d\}D+\{(2^w-w)\}A$ for $w=2,3,4,5$ in HPB's paper [18]. This means that sABS needs one more point addition than sNZ in the precomputation stage.

The evaluation stage for both HPB's comb methods costs $d-1$ point additions and $d-1$ doublings. The last stage after the original comb method costs one doubling and one subtraction. Therefore the total cost of sNZ is $(w-1)d+(d-1)+1=wd$ doubling operations and $(2^w-w-1)+(d-1)+1=2^w-w+d-1$ adding operations. sABS costs $\{wd\}D+\{(2^w-w+d)\}A$. Compared with the original comb method Alg. 1, sABS stores about half of precomputed points as that in the original comb method, but the time cost in the precomputation stage is a little higher due to the fact that all bits in sABS are set to either 1 or -1.

FZXL's Comb Methods. Feng, Zhu, Xu, and Li (FZXL) [19] proposed another comb method referred to as the *signed LSB-Set* (sLSBS) comb method in this paper and its variations which are more efficient than the original comb method. In sLSBS, every odd scalar k is transformed into a representation of bit-columns $\{\mathbb{K}_i\}$ with the following properties: for each bit-column $[b_{w-1},\cdots,b_1,b_0]$, the least significant bit b_0 is either 1 or -1, and the rest bits b_i is either 0 or has the same sign as b_0, $0<i<w$. In other words, $\mathbb{K}_i=\pm[c_{w-1},\cdots,c_2,c_1,1]$, where $c_i=0$ or 1 for $0<i<w$. By adding dummy operations, sLSBS is easily modified to be SPA-resistant. Both versions store 2^{w-1} precomputed points with the time cost in the precomputation stage as $\{(w-1)d\}D+\{(2^{w-1}-1)\}A$. sLSBS requires $(d-1)D+(d-\frac{1}{2})A$ in the evaluation stage and the total cost is $(wd-1)D+(2^{w-1}+d-\frac{3}{2})A$. The corresponding values for the SPA-resistant version are $dD+dA$ and $wdD+(2^{w-1}+d-1)A$ [19], respectively.

The value d used in sLSBS or its SPA-resistant counterpart is equal to $\lceil\frac{n+1}{w}\rceil$ instead of $\lceil\frac{n}{w}\rceil$ used by other comb methods, which results in d one larger than that used in other comb methods when n, the number of bits of k, is divisible by w. FZXL proposed several methods to deal with this issue while maintaining computational efficiency. Details can be found in [19]. In this paper, we compare our proposed comb methods with only sLSBS and its SPA-resistant counterpart.

4 Signed MSB-Set Comb Method

4.1 Recoding Algorithm

Like the aforementioned comb methods, our approach is also to represent a scalar k with a set of signed nonzero bit-columns $\{\mathbb{K}'_i\equiv[K'^{w-1}_i,\cdots,K'^{1}_i,K'^{0}_i]\neq 0\}$. The major difference is that every $\mathbb{K}'_i$ generated by our novel recoding method is a signed MSB-set integer. More specifically, our recoding scheme generates $K'^{w-1}_i\in\{1,\bar{1}\}$ and $K'^{j}_i\in\{0,K'^{w-1}_i\}$, $0\leqslant j<w-1$ for each bit-column $\mathbb{K}'_i$. As shown later in this paper, a major advantage of our recoding method over the original fixed-base comb recoding method is that the precomputation stage

needs to calculate and store only half of the points. The detail of our recoding algorithm is described next for a window width $w \geqslant 2$.

The recoding algorithm first partitions a binary representation of a scalar k into w binary strings K^j of d bits long for each, $0 \leqslant j < w$, with 0 possibly padded on the left. Then it converts in Steps 3 to 5 each bit of the highest d bits to either 1 or $\bar{1}$ in by exploiting the fact that $1 \equiv 1\bar{1}\bar{1}\cdots\bar{1}$. In other words, each bit $K_r'^{w-1}$, $0 \leqslant r < d$, in K'^{w-1} is either 1 or $\bar{1}$. The rest of the recoding algorithm processes each bit from the least significant bit towards the $\{(w-1)d-1\}^{th}$ bit. If the current i^{th} bit b_i is 1 and has a sign different from that of the most significant bit $b'_{(i \bmod d)+(w-1)d}$ in the same bit-column $\mathbb{K}'_{i \bmod d}$, the current bit is set to $\bar{1}$ and the next higher corresponding bit is added by 1 to keep the value of k unchanged. This process generates wd bits $\{b'_i\}$ and a δ to represent an n-bit integer k. Due to length limitation, the following theorems are given without proof.

Theorem 1. *Given a scalar k, Alg. 2 outputs a $\delta \in \{0, \pm 1\}$, a sequence of bits $\{b'_i\}$, and bit-columns $\{\mathbb{K}'_r \equiv [K_r'^{w-1}, \cdots, K_r'^1, K_r'^0]\}$ such that $k = \delta \cdot 2^{(w-1)d} + \sum_{i=0}^{wd-1} b'_i 2^i$ and for each $\mathbb{K}'_r$, $K_r'^{w-1} \in \{1, -1\}$ and $K_r'^j \in \{0, K_r'^{w-1}\}$, where $0 \leqslant j < w-1$ and $0 \leqslant r < d$, and $K_r'^j \equiv b'_{jd+r}$.*

Theorem 2. *δ has a probability of $\frac{1}{2}$ to be zero when k is an integer in $[0, 2^n)$ with uniform distribution.*

Algorithm 2. *Signed MSB-Set Comb Recoding Algorithm ($d = \lceil \frac{n}{w} \rceil$).*

Input: *An n-bit integer $k > 0$ and a window width $w \geqslant 2$.*
Output: *$k = \delta \cdot 2^{(w-1)d} + \sum_{i=0}^{wd-1} b'_i 2^i \equiv \delta \cdot 2^{(w-1)d} + K'^{w-1} || \cdots || K'^1 || K'^0$, where each K'^j is a binary string of d bits long and $\delta \in \{0, \pm 1\}$. Let $K_r'^j$ denote the r^{th} bit of K'^j, i.e., $K_r'^j \equiv b'_{jd+r}$. Define bit-column $\mathbb{K}'_r \equiv [K_r'^{w-1}, \cdots, K_r'^1, K_r'^0]$. The output satisfies $K_r'^{w-1} \in \{1, -1\}$ and $K_r'^j \in \{0, K_r'^{w-1}\}$ for $0 \leqslant j < w-1$ and $0 \leqslant r < d$.*

```
1. Padding with 0 on the left if necessary to form a wd-bit representation
   $k = \sum_{i=0}^{wd-1} b_i 2^i$ with $b_i \in \{0, 1\}$.
2. Set $b'_{(w-1)d} = 1$, $\delta = b_{(w-1)d} - 1$ and $e = \sum_{i=0}^{(w-1)d-1} b_i 2^i$.
3. For $i = (w-1)d + 1$ to $wd - 1$ by 1 do:
4.     if $b_i = 1$ then set $b'_i = 1$,
5.     if $b_i = 0$ then set $b'_i = 1$ and $b'_{i-1} = \bar{1}$.
6. For $i = 0$ to $(w-1)d - 1$ by 1 do
7.     if $e$ is odd and $b'_{(i \bmod d)+(w-1)d} = \bar{1}$, then set $b'_i = \bar{1}$ and $e = \lceil \frac{e}{2} \rceil$
8.     else set $b'_i = e \bmod 2$, and $e = \lfloor \frac{e}{2} \rfloor$
9. $\delta = \delta + e$
```

4.2 Signed MSB-Set Comb Methods

By applying the recoding method in Sect. 4.1, we have the comb method Alg. 3. If the most significant bit of $\mathbb{K}'_i$ is $\bar{1}$, we have $\mathbb{K}'_i = -|\mathbb{K}'_i|$. In this case, Step 6 in Alg. 3 actually executes $Q = Q - |\mathbb{K}'_i| P$.

Algorithm 3. *Signed MSB-Set Comb Method ($d = \lceil \frac{n}{w} \rceil$).*

```
Input: A point $P$, an integer $k > 0$, and a window width $w \geqslant 2$.
Output: $Q = kP$.
Precomputation Stage:
1. Compute $[1, b_{w-2}, \cdots, b_1, b_0]P$ for all $(b_{w-2}, \cdots, b_1, b_0) \in \{0,1\}^{w-1}$.
   (Note that $[1, 0, \cdots, 0, 0] = 2^{(w-1)d}$.)
2. Apply Alg. 2 to $k$ to compute the corresponding bit-columns $\mathbb{K}'_0, \mathbb{K}'_1, \cdots, \mathbb{K}'_{d-1}$
   and $\delta$.
3. $Q = \mathcal{O}$.
Evaluation Stage:
4. For $i = d-1$ to $0$ by $-1$ do:
5.     $Q = 2Q$,
6.     $Q = Q + \mathbb{K}'_i P$.
7. Return $Q = Q + \delta \cdot 2^{(w-1)d} P$ (i.e., return $Q = Q + \delta \cdot [1, 0, \cdots, 0, 0]P$).
```

Alg. 3 is not an SPA-resistant comb method. SPA is able to detect if δ is zero or not in Step 7 of Alg. 3. Since $\mathbb{K}_i \neq 0$ for all i, the operations in the *for* loop of Alg. 3 are a sequence of alternative point doubling (D) and point addition (A), $DADA \cdots DADA$. By inserting potential dummy operations after the *for* loop, we can easily convert the above SPA-nonresistant method to an SPA-resistant method, as described in the following algorithm.

Algorithm 4. *SPA-Resistant Signed MSB-Set Comb Method ($d = \lceil \frac{n}{w} \rceil$).*

```
Input: A point $P$, an integer $k > 0$, and a window width $w \geqslant 2$.
Output: $Q = kP$.
Precomputation Stage:
1. Compute $[1, b_{w-2}, \cdots, b_1, b_0]P$ for all $(b_{w-2}, \cdots, b_1, b_0) \in \{0,1\}^{w-1}$.
2. Apply Alg. 2 to $k$ to compute the corresponding bit-columns $\mathbb{K}'_0, \mathbb{K}'_1, \cdots, \mathbb{K}'_{d-1}$
   and $\delta$.
3. $Q_0 = \mathcal{O}$.
Evaluation Stage:
4. For $i = d-1$ to $0$ by $-1$ do:
5.     $Q_0 = 2Q_0$,
6.     $Q_0 = Q_0 + \mathbb{K}'_i P$.
7. Set $Q_1 = Q_0 - (-1)^{b_{(w-1)d}} \cdot [1, 0, \cdots, 0, 0]P$.
8. Return $Q_{|\delta|}$.
```

In Steps 7–8 of Alg. 4, we have exploited the fact that $\delta = b_{(w-1)d} - 1 + e_{(w-1)d}$ from Alg. 2, where $b_{(w-1)d}$ and $e_{(w-1)d}$ are in $\{0, 1\}$. This fact implies that $b_{(w-1)d}$ must be 0 if δ is -1, and $b_{(w-1)d}$ must be 1 if δ is 1.

4.3 Security Against Power Analysis

Security of our proposed SPA-resistant point multiplication method Alg. 4 is discussed in this section. We first consider its security against SPA, and then describe how to convert the method to resist DPA, second-order DPA, and other side channel attacks.

Like other SPA-resistant methods [17, 15, 16], Alg. 4 exploits the fact that point subtraction is virtually the same as point addition for power analysis. It performs one point addition (or point subtraction) and one doubling in each iteration of the loop in calculating point multiplication. There is always one point addition (or point subtraction) in Step 7. This means that the same sequence is executed no matter what value a scalar k is. Therefore SPA cannot extract any information about the secret k by examining the power consumption in execution of Alg. 4's point multiplication. In other words, our SPA-resistant comb method Alg. 4 is really SPA-resistant.

An SPA-resistant method is not necessarily resistant to DPA attacks, as shown by other SPA-resistant point multiplication methods [17,15,16]. This is also true for our SPA-resistant method Alg. 4. Typical measures such as randomization projective coordinates or random isomorphic curves can be used to convert Alg. 4 into a DPA-resistant method.

The aforementioned randomization measures may not be enough to resist the second-order DPA attack proposed by Okeya and Sakurai [6]. This second order attack exploits the correlation between power consumption and hamming weight of the loaded data to determine which $\mathbb{K}'_i$ is loaded. To thwart this second-order DPA attack, we can use the same scheme proposed in [17] to protect HPB's methods – to randomize all precomputed points after getting the point in the table so that there is no fixed hamming weight.

Goubin [7] recently proposed a refined DPA attack on many randomization schemes. This attack employs special points with one of coordinates being zero. To deal with Goubin's DPA attack, a simple approach is to choose elliptic curves

$$E : y^2 = x^3 + ax + b$$

defined over F_p $(p > 3)$ with b not being a quadratic residue modulo p, and to reject any point $(x, 0)$ as an input point in applications of our proposed SPA-resistant method. If the cardinality $\#E(F_p)$ is a big prime number, points $(x, 0)$ cannot be eligible input points since they are not on elliptic curves. Another more powerful attack, the Zero-value Point Attack proposed in [8] also requires certain prerequisite conditions for the elliptic curve to be used, although the conditions are weaker than Goubin's attack [7]. Careful selection of the elliptic curve can get rid of these security threats.

4.4 Efficiency

Both of our comb methods Algs. 3–4 require storage of 2^{w-1} points. In the precomputation stage of our comb methods, $2^d P, 2^{2d} P, \cdots, 2^{(w-1)d} P$ are first calculated. This costs $(w-1)d$ point doublings. Then all possible combinations $[1, b_{w-2}, \cdots, b_1, b_0]P$ with $(b_{w-2}, \cdots, b_1, b_0) \in \{0, 1\}^{w-1}$ are calculated in the same way as the precomputation stage for Alg. 1, which costs $2^{w-1} - 1$ point additions. The total cost of our comb methods in the precomputation stage is

Table 1. Comparison of space and average time costs for the SPA-nonresistant comb methods

	Original Comb	sLSBS Comb	Alg. 3
d	$\lceil \frac{n}{w} \rceil$	$\lceil \frac{n+1}{w} \rceil$	$\lceil \frac{n}{w} \rceil$
Storage	$2^w - 1$	2^{w-1}	2^{w-1}
Pre-Stage	$(w-1)dD$ $(2^w - w - 1)A$	$(w-1)dD$ $(2^{w-1} - 1)A$	$(w-1)dD$ $(2^{w-1} - 1)A$
Eva-Stage	$(d-1)D$ $\frac{2^w-1}{2^w}(d-1)A$	$(d-1)D$ $(d-\frac{1}{2})A$	$(d-1)D$ $(d-\frac{1}{2})A$
Total Cost	$(wd-1)D$ $(2^w - w - 1 + \frac{2^w-1}{2^w}(d-1))A$	$(wd-1)D$ $(2^{w-1} + d - \frac{3}{2})A$	$(wd-1)D$ $(2^{w-1} + d - \frac{3}{2})A$

therefore $\{(w-1)d\}D + \{2^{w-1} - 1\}A$. The time costs of our comb methods in the evaluation stage vary a little due to the post-processing after the *for* loop. Assume that the scalar k is uniformly distributed, then the average cost in the evaluation stage is $(d-1)D + (d-\frac{1}{2})A$ for Alg. 3 and $(d-1)D + dA$ for Alg. 4.

We would like to first compare our comb method Alg. 3 with the other fixed-base comb methods without considering SPA-resistance. Table 1 lists the space and time costs for the original comb method Alg. 1, sLSBS [19], and Alg. 3. Compared with the original comb method Alg. 1, our comb method Alg. 3 stores 2^{w-1} points, which is about half of $2^w - 1$, the number of points stored by the original comb method. In addition, Alg. 3 saves $2^{w-1} - w$ point additions in the precomputation stage. The evaluation stage has a similar time cost for both methods, as shown in Table 1. If we want to maintain about the same storage space for pre-computed points, our method Alg. 3 can choose the value of w as $w = w_1 + 1$, one larger than the value $w = w_1$ used in the original comb method, resulting in a similar storage space (2^{w_1} v.s. $2^{w_1} - 1$) as the original comb method Alg. 1 yet with much faster computation in both precomputation and evaluation stages, thanks to smaller d used in our methods. Compared with sLSBS, d changes from $\lceil \frac{n+1}{w} \rceil$ in sLSBS to $\lceil \frac{n}{w} \rceil$ in Alg. 3. When n is not divisible by w, both methods have the same value of d, resulting in the same time cost. In the case that n is divisible by w, Alg. 3 uses a d which is one smaller than that used in sLSBS, resulting in smaller time costs in both precomputation and evaluation stages. To deal with the problem, FZXL proposed several different schemes to ensure execution efficiency. Our proposed comb methods handle the issue in a nice and uniform manner.

Let us now compare our SPA-resistant comb method Alg. 4 with other SPA-resistant comb methods. Table 2 lists the space and time costs for the two HPB methods, sLSBS's SPA-resistant counterpart, and our SPA-resistant comb method Alg. 4. All the comb methods except sNZ store 2^{w-1} pre-computed points, while sNZ stores $2^w - 1$ pre-computed points which is about twice the number of stored points in other comb methods. Our Alg. 4 executes one less point doubling in the evaluation stage than the other three SPA-resistant comb methods, and requires much less point additions in the precomputation stage than both of HPB's methods. Alg. 4 shows the same advantage over sLSBS's

Table 2. Comparison of space and average time costs for SPA-resistant comb methods

	sNZ Comb	sABS Comb	sLSBS[a] Comb	Alg. 4
d	$\lceil \frac{n}{w} \rceil$	$\lceil \frac{n}{w} \rceil$	$\lceil \frac{n+1}{w} \rceil$	$\lceil \frac{n}{w} \rceil$
Storage	$2^w - 1$	2^{w-1}	2^{w-1}	2^{w-1}
Pre-Stage	$(w-1)dD$ $(2^w - w - 1)A$	$(w-1)dD$ $(2^w - w)A$	$(w-1)dD$ $(2^{w-1} - 1)A$	$(w-1)dD$ $(2^{w-1} - 1)A$
Eva-Stage	dD dA	dD dA	dD dA	$(d-1)D$ dA
Total Cost	wdD $(2^w - w + d - 1)A$	wdD $(2^w - w + d)A$	wdD $(2^{w-1} + d - 1)A$	$(wd-1)D$ $(2^{w-1} + d - 1)A$

[a]SPA-resistant version

SPA-resistant counterpart as Alg. 3 over sLSBS, due to smaller d used in Alg. 4 when n is divisible by w. Table 2 shows that our Alg. 4 is the most efficient SPA-resistant comb method.

5 Conclusion

In this paper, we proposed a novel comb recoding algorithm to convert an integer to a representation with a set of signed MSB-set nonzero comb bit-columns. Using this recoding algorithm, we presented a signed MSB-set comb method and an SPA-resistant comb method to calculate point multiplication for ECC. Security of the proposed SPA-resistant comb method and comparison of the proposed comb methods with other comb methods were also discussed in the paper. Our comb methods are among the most efficient comb methods in terms of the number of precomputed points and computational complexity. Combined with randomization techniques and certain precautions in selecting elliptic curves and parameters, our proposed SPA-resistant comb methods can thwart all side-channel attacks.

References

1. I. F. Blake, G. Seroussi, and N. P. Smart, *Elliptic Curves in Cryptography*, Cambridge Univ. Press, 1999.
2. C. Lim and P. Lee, "More Flexible Exponentiation with Precomputation," *Advances in Cryptology – CRYPTO'94,* LNCS 839, pp. 95–107, Springer-Verlag, 1994.
3. P. C. Kocher, "Timing Attacks on Implementations of Diffe-Hellman, RSA, DSS and Other Systems," *Advances in Cryptology – CRYPTO'96*, LNCS 1109, pp. 104–113, Springer-Verlag, 1996.
4. P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," *Advances in Cryptology – CRYPTO'99*, LNCS 1666, pp. 388–397, Springer-Verlag, 1999.
5. J. S. Coron, "Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems," *Cryptographic Hardware and Embedded Systems – CHES'99,* LNCS 1717, pp. 292–302, Springer-Verlag, 1999.

6. K. Okeya and K. Sakurai, "A Second-Order DPA Attack Breaks a Window-Method Based Countermeasure against Side Channel Attacks," *Proc. 5th Intl. Conf. on Information Security*, LNCS 2433, pp. 389–401, Springer-Verlag, 2002.
7. L. Goubin, "A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems," *Public Key Cryptography – PKC'2003*, LNCS 2567, pp 199–211, Springer-Verlag, 2003.
8. T. Akishita and T. Takagi, "Zero-Value Point Attacks on Elliptic Curve Cryptosystem," *Information Security Conference – ISC'2003*, LNCS 2851, pp 218–233, Springer-Verlag, 2003.
9. P. Y. Liardet and N. P. Smart, "Preventing SPA/DPA in ECC Systems Using the Jacobi Form," *Cryptographic Hardware and Embedded Systems – CHES'2001*, LNCS 2162, pp. 391–401, Springer-Verlag, 2001.
10. M. Joye and J. J. Quisquater, "Hessian Elliptic Curves and Side-Channel Attacks," *Cryptographic Hardware and Embedded Systems – CHES'2001*, LNCS 2162, pp. 402–410, Springer-Verlag, 2001.
11. K. Okeya, H. Kurumatani, and K. Sakurai, "Elliptic Curves with the Montgomery-Form and Their Cryptographic Applications," *Public Key Cryptography–PKC'2000*, LNCS 1751, pp. 238–257, Springer-Verlag, 2000.
12. W. Fischer, C. Giraud, E. W. Knudsen, and J.-P. Seifert, "Parallel Scalar Multiplication on General Elliptic Curve over F_p Hedged against Non-Differential Side-Channel Attacks," *IACR, Cryptography ePrint Archieve 2002/007*, http://eprint.iacr.org/2002/007, 2002.
13. T. Izu, and T. Takagi, "A Fast Parallel Elliptic Curve Multiplication Resistant against Side Channel Attacks," *Public Key Cryptography (PKC 2002)*, LNCS 2274, pp. 280–296, 2002.
14. E. Brier and M. Joye, "Weierstrass Elliptic Curves and Side-Channel Attacks," *Public Key Cryptography (PKC2002)*, LNCS 2274, pp. 335–345, 2002.
15. B. Möller, "Securing Elliptic Curve Point Multiplication against Side-Channel Attacks, Addendum: Efficiency Improvement," http://www.informatik.tudarmstadt.de/TI/Mitarbeiter/moeller/ecc-scaisc01.pdf, 2001.
16. K. Okeya and T. Takagi, "A More Flexible Countermeasure against Side Channel Attacks Using Window Method" *cryptographic Hardware and Embedded Systems – CHES'2003*, LNCS 2779, pp. 397–410, 2003.
17. M. Hedabou, P. Pinel, and L. Bébéteau, "A Comb Method to Render ECC Resistant against Side Channel Attacks," http://eprint.iacr.org/2004/342.pdf, 2004.
18. M. Hedabou, P. Pinel, and L. Bébéteau, "Countermeasures for Preventing Comb Method Against SCA Attacks," *Information Security Practise and Experience Conference, ISPEC'2005*, LNCS 3439, pp, 85-96, Springer-Verlag, 2005.
19. M. Feng, B. Zhu, M. Xu and S. Li, "Efficient Comb Methods for Elliptic Curve Point Multiplication Resistant to Power Analysis," http://eprint.iacr.org/2005/222.
20. B. Chevallier-Mames, M. Ciet, and M. Joye, "Low-Cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity," *IEEE Transaction on Computers*, vol. 53, no. 6, pp. 760–768, June 2004.
21. M. Joye and C. Tymen, "Protections against Differential Analysis for Elliptic Curve Cryptography – An Algebraic Approach," *Cryptographic Hardware and Embedded Systems – CHES'2001*, LNCS 2162, pp. 377–390, Springer-Verlag, 2001.